Brain Tumor Classification using Transfer Learning

Project ID: 26135

B.Tech. Project Report

submitted for fulfillment of

the requirements for the

Degree of Bachelor of Technology

Under Biju Pattnaik University of Technology

Submitted By

Debashish Senapati **ROLL NO: CSE201910457**

Sundeep Panigrahi **ROLL NO: CSE201940263**

2022 – 2023

Under the guidance of

Prof. Charulata Palai

NIST INSTITUTE OF SCIENCE & TECHNOLOGY (Autonomous)

Institute Park, Palur Hills, Berhampur, Odisha – 761008, India

ABSTRACT

A brain tumor is a distorted tissue wherein cells replicate rapidly and indefinitely, with no control over tumor growth. Detection of these tumor cells becomes challenging. Deep learning has been argued to have the potential to overcome the challenges associated with detecting and intervening in brain tumors. In the case of brain tumors, early disease detection can be achieved effectively using reliable advanced A.I. and Neural Network classification algorithms. Here, the Convolutional neural network will be used to classify the brain tumor.

To date, researchers have designed lots of brilliant deep convolutional neural networks (D-CNNs). However, most of the existing deep convolutional neural networks are trained with large datasets. It is quite rare for small datasets to take advantage of deep convolutional neural networks because of overfitting during the implementation of those models. In this study, we use Visual Geometry Group (VGG 16) for brain tumor detection, propose a modified Convolutional Neural Network (CNN) model and set the parameters to train the model for tumor classification. We test our model using Brain MRI Images dataset to measure the Training Accuracy and Validation Accuracy of our proposed model and measure the performance of the model using various metrics such as Accuracy, Precision, Recall, F1 score, Sensitivity, Specificity, Precision-Recall Curve, Confusion matrix Receiver Operating Characteristic (ROC) curve.

In addition, the study addresses a practical aspect by evaluating the system with fewer training samples. The observations of the study imply that transfer learning is a useful technique when the availability of medical images is limited.

Keywords: Brain Tumor, Convolutional Neural Network, VGG16

ACKNOWLEDGEMENT

We would like to take this opportunity to thank all those individuals whose invaluable contribution in a direct or indirect manner has gone into the making of this project a tremendous learning experience for me.

It is our proud privilege to epitomize my deepest sense of gratitude and indebtedness to my faculty guide, **Prof. Charulata Palai** for her valuable guidance, keen and sustained interest, intuitive ideas and persistent endeavor. Her guidance and inspirations enabled me to complete my report work successfully.

We give our sincere thanks to **Prof. Rajesh Ku. Dash (**B.Tech Project Coordinator) for giving me the opportunity and motivating me to complete the project within stipulated period of time and providing a helping environment.

We acknowledge with immense pleasure the sustained interest, encouraging attitude and constant inspiration rendered by **Prof. (Dr.) Sukant K. Mohapatra (Chairman), Dr. Priyadarshi Tripathy (Principal), Dr. Sudhir Ranjan Pattanaik (H.O.D CSE), and Dr. Susmita Mahato (Department Project Coordinator)**. Their continued drive for better quality in everything that happens at N.I.S.T. and selfless inspiration has always helped us to move ahead.

Debashish Senapati

Roll No: CSE201910457

Sundeep Panigrahi

Roll No: CSE201940263

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The brain is one of the most important organs of our body. It controls our actions and the functions of the body. It is made up of more than 100 billion nerves that communicate with each other to carry out the daily tasks of our body. A brain tumor is an abnormal growth of cells in the brain or skull.

Medical imaging technique is used to create visual representation of the inside of the human body for medical purposes, and this technology can be used for non-invasive diagnoses. The different types of medical imaging technologies are based on non-invasive approaches such as: MRI, CT scan, ultrasound and X-ray. Compared to other medical imaging technologies, Magnetic Resonance Imaging (MRI) is mainly used, which provides higher contrast images of the brain and cancerous tissue. Therefore, classification of brain tumors can be done based on MRI images **[2]**.

In the medical field, MRI image processing for brain tumor detection and classification is challenging due to the complexity and different forms of tumors. If the detection of brain tumors can be done quickly and accurately, it can significantly support the initial treatment of patients, i.e. it can provide remedial measures to be taken as the first step towards recovery. Early detection of brain tumors is therefore of paramount importance, and with advances in medical science and technology, this is now possible.

With the help of Artificial Intelligence and Deep Learning software can be developed that can detect tumors and classify them into different types. Image processing and transfer learning are used to propose a system for classifying brain tumors. Transfer learning allows the use of a pre-trained CNN model that was actually developed for another related application. In the field of medical imaging, artificial intelligence and digital image processing, the Convolutional Neural Network (CNN) has had a major impact. Classifying brain tumors into subtypes is a difficult research problem.

The challenges associated with it are due to the following factors:

1. Brain tumors have wide variations in shape, size and intensity.
2. Tumors of different pathological types may have a similar appearance.
3. Among all brain tumors, gliomas, meningiomas and pituitary tumors have the highest incidence rates **[6]**.

Here we present an accurate and automatic classification system developed for brain tumor detection and classification. The implementation uses a deep CNN model with transfer learning for feature extraction from MRI images of the brain. The proposed algorithm provides acceptable results even with a smaller number of training samples.

2. LITERATURE SURVEY

The Study Brain Tumor Classification Using CNN and VGG16 Model by Anushka Singh, Rajeshwari Deshmukh, Riya Jha, Nishi Shahare, Sonam Verma, Prof.Ajinkya Nilawar Proposed a deep learning model for classifying tumors using CNN as a classifier. Based on their result, an accuracy of above 93% along with high precision, recall and F-score was achieved. As we know VGG16 is a pre-trained model and using its architecture was quite helpful for the CNN architecture. It has a good performance for differentiating the types of tumor.

Brain Tumor Analysis Using Deep Learning and VGG-16 Ensembling Learning Approaches by Ayesha Younis,Li Qiang, Charles Okanda Nyatega, Mohammed Jajere Adamu and Halima Bello Kawuwa tried to meet such needs by bringing a novel solution for the detection of brain tumors in MRI images with greater precision. The proposed model integrated deep learning and transfer learning models to achieve a remarkable accuracy rate. the optimization of training models was increased to reduce the need for high computational power. A CNN was built to detect brain tumors using MRI scans of the brain automatically. The network could be trained for faster and more convenient training using a pre-trained VGG 16 model.

Brain tumor classification using deep CNN features via transfer learning by S. Deepak, P.M. Ameer presents an accurate and fully automatic system, with minimum pre-processing, for brain tumor classification. The proposed system applied the concept of deep transfer learning to extract features from brain MRI images. The system recorded the best classification accuracy compared to all the related works. The observations of the study imply that transfer learning is a useful technique when the availability of medical images is limited.

Brain Tumor Detection and Classification Using Convolutional Neural Network and Deep Neural Network by C L Choudhury, Chandrakanta Mahanty, Raghvendra Kumar, B K Mishra a new system based on CNN, which discriminates between the Brain MRI images to mark them as tumorous or not. The model achieved the accuracy of 96.08%, with a f-score of 97.3. The model has CNN with 3 layers and requires very few steps of pre-processing to produce the results in 35 Epochs.

3. TRANSFER LEARNING

3.1 What is Transfer Learning?

Transfer learning (TL) is a research problem in Machine Learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. It refers to the situation whereby what has been learnt in one setting is exploited to improve optimization in another setting. These algorithms are used to create and maintain patterns that are formed.

3.2 Necessity for transfer learning

Low-level features learned for task A should be beneficial for learning of model for task B. Nowadays, it is very hard to see people training a whole convolutional neural network from scratch, and it is common to use a pre-trained model trained on a variety of images in a similar task, e.g. models trained on ImageNet (1.2 million images with 1000 categories), and use features from them to solve a new task.

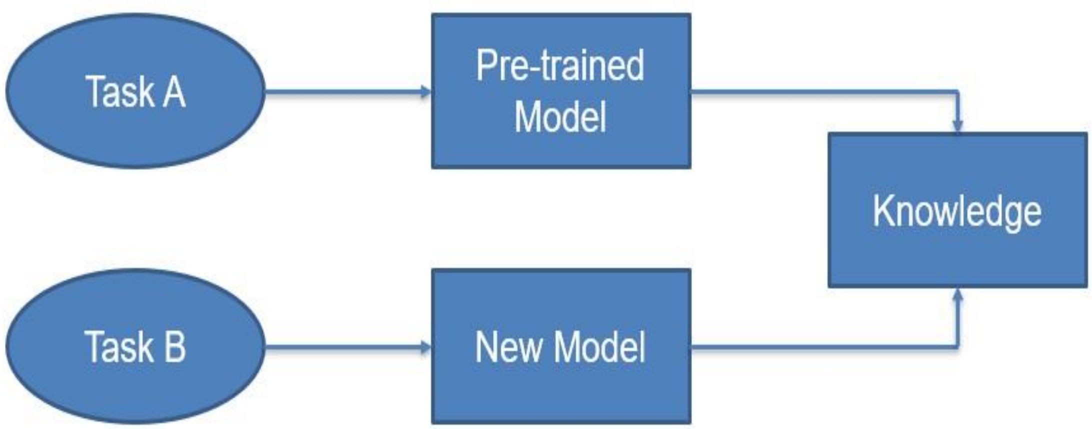


Figure 3.1: Block diagram of Transfer Learning

When dealing with transfer learning, we come across a phenomenon called freezing of layers. A layer, it can be a CNN layer, hidden layer, a block of layers, or any subset of a set of all layers, is said to be fixed when it is no longer available to train. Hence, the weights of feezed layers will not be updated during training. While layers that are not feezed follows regular training procedure. When we use transfer learning in solving a problem, we select a pre-trained model as our base model. Now, there are two possible approaches to use knowledge from the pre-trained model. First way is to freeze a few layers of pre-trained model and train other layers on our new dataset for the new task. Second way is to make a new model, but also take out some features from the layers in the pre-trained model and use them in a newly created model. In both cases, we take out some of the learned features and try to train the rest of the model. This makes sure that the only feature that may be same in both of the tasks is taken out from the pre-trained model, and the rest of the model is changed to fit new dataset by training.

4. DEEP LEARNING

4.1 What is Deep Learning?

Deep learning is a branch of machine learning which is completely based on ANN. In deep learning, we don't need to explicitly program everything. The concept of deep learning is not new. It has been around for a couple of years now. It's on hype nowadays because earlier we did not have that much processing power and a lot of data.

Formal Definition: Deep learning is a particular kind of machine learning that achieves great power and flexibility by learning to represent the world as a nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones.

4.2 How Deep Learning works?

Neural networks are layers of nodes, much like the human brain is made up of neurons. Nodes within individual layers are connected to adjacent layers. The network is said to be deeper based on the number of layers it has. A single neuron in the human brain receives thousands of signals from other neurons. A heavier weighted node will exert more effect on the next layer of nodes. The final layer compiles the weighted inputs to produce an output. Deep learning systems require powerful hardware because they have a large amount of data being processed and involves several complex mathematical calculations. Even with such advanced hardware, however, training a neural network can take weeks. Deep learning systems require large amounts of data to return accurate results; accordingly, information is fed as huge data sets. When processing the data, artificial neural networks are able to classify data with the answers received from a series of binary true or false questions involving highly complex mathematical calculations. For example, a facial recognition program works by learning to detect and recognize edges and lines of faces, then more significant parts of the faces, and, finally, the overall representations of faces. Over time, the program trains itself, and the probability of correct answers increases. In this case, facial recognition program will accurately identify faces with time.

5. NEURAL NETWORK

5.1 What is Neural Networks?

These are inspired by the most complex object in the universe – the human brain. The human brain is made up of something called Neurons. A neuron is the most basic computational unit of any neural network, including the brain. A neural network is a system or hardware that is designed to operate like a human brain.

Neural networks can perform the following tasks:

- Translate text
- Identify faces
- Recognize speech
- Read handwritten text
- Control robots
- And a lot more

5.2 Working of Neural Networks?

A neural network is usually described as having different layers. The first layer is the input layer, it picks up the input signals and passes them to the next layer. The next layer does all kinds of calculations and feature extractions—it's called the hidden layer. Often, there will be more than one hidden layer. And finally, there's an output layer, which delivers the final result.

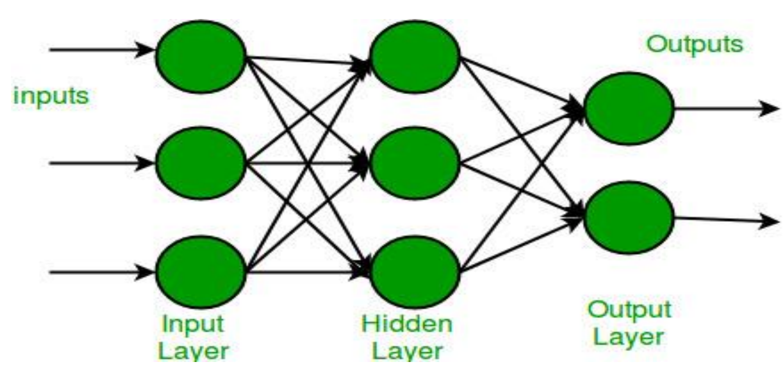


Figure 5.1: Neural Network

The weighted sum of the inputs is fed as input to the activation function, to decide which nodes to fire for feature extraction. As the signal flows within the hidden layers, the weighted sum of inputs is calculated and is fed to the activation function in each layer to decide which nodes to fire. There are different types of activation functions, such as:

5.2.1 Sigmoid Function

The sigmoid function is used when the model is predicting probability.

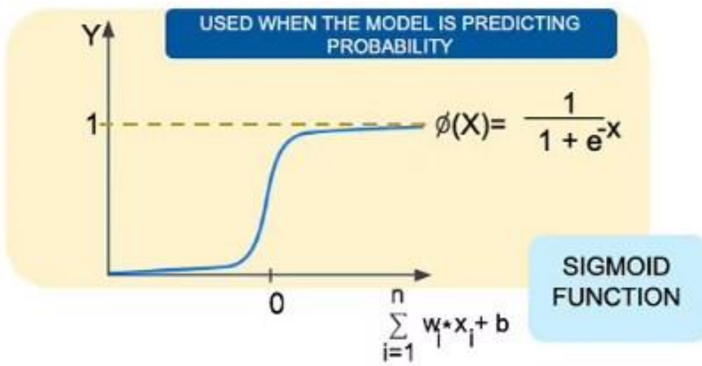


Figure 5.2: Sigmoid Function

5.2.2 Threshold Function

The threshold function is used when you don’t want to worry about the uncertainty in the middle.

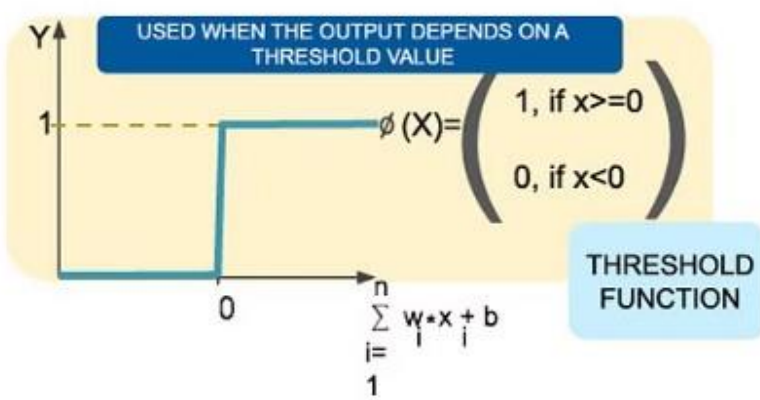


Figure 5.3: Threshold Function

5.2.3 ReLU (Rectified Linear Unit) Function

The ReLU function gives the value but says if it's over 1, then it will just be 1, and if it's less than 0, it will just be 0. The ReLU function is most commonly used these days.

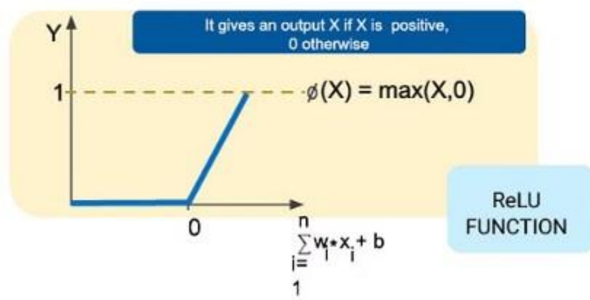


Figure 5.4: ReLU Function

5.2.4 Softmax Function

The softmax function transforms the input vector into a probability distribution over the K classes, which can then be used to predict the class with the highest probability. Specifically, the softmax function applies the exponential function to each element of the input vector, which ensures that the output values are positive. It then divides each element by the sum of all the exponential values, which ensures that the output values sum up to 1.

6. DATASET DESCRIPTION

The pre-trained model VGG16 used ImageNet, a dataset of over 14 million images belonging to over 1000 classes and over 22,000 categories. The ImageNet dataset consists of roughly 1.2 million training images, 50,000 validation images and 150,000 test images. Since this dataset consists of variable resolution images, all these images have been down-sampled to a fixed resolution of 256x256 patch.

The dataset used in this study was “Brain MRI Images”. [Figure-6] depicts the structure of the Brain MRI Images dataset. The employed dataset included three distinct and well-known kinds of brain cancer such as Glioma, Pituitary, Meningioma and another class having No Tumor. The models were trained and tested using MRI dataset that included around 3000 brain tumor images of 4 different class. The tumor dataset was submitted for pre-processing and encoded.

- Glioma Tumor - encoded as 0
- Pituitary Tumor - encoded as 1
- No Tumor - encoded as 2
- Meningioma Tumor - encoded as 3

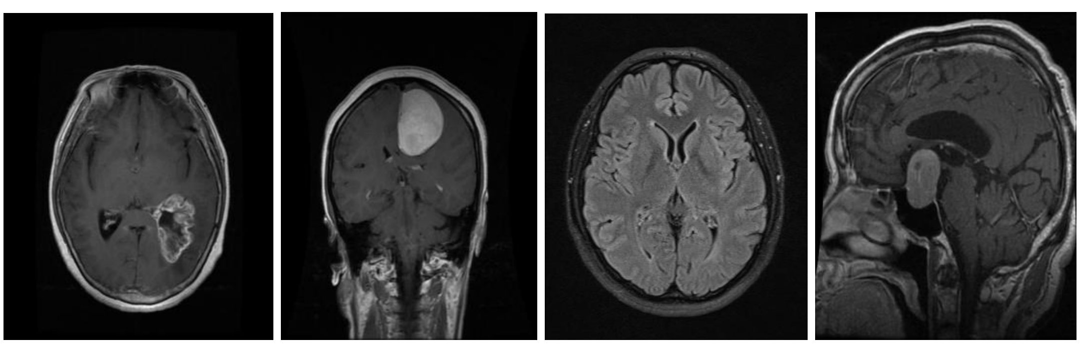


Figure 6.1: Brain MRI Images

7. PROPOSED SYSTEM

7.1 CNN Model

7.1.1 What is CNN?

Convolutional Neural Network (CNN) is used to classify the data. It has completely dominated the machine vision space in recent years. A CNN consists of an input layer, output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers and normalization layers (ReLU). Additional layers can be used for more complex models. Examples of a typical CNN can be seen in [Figure 7.1].

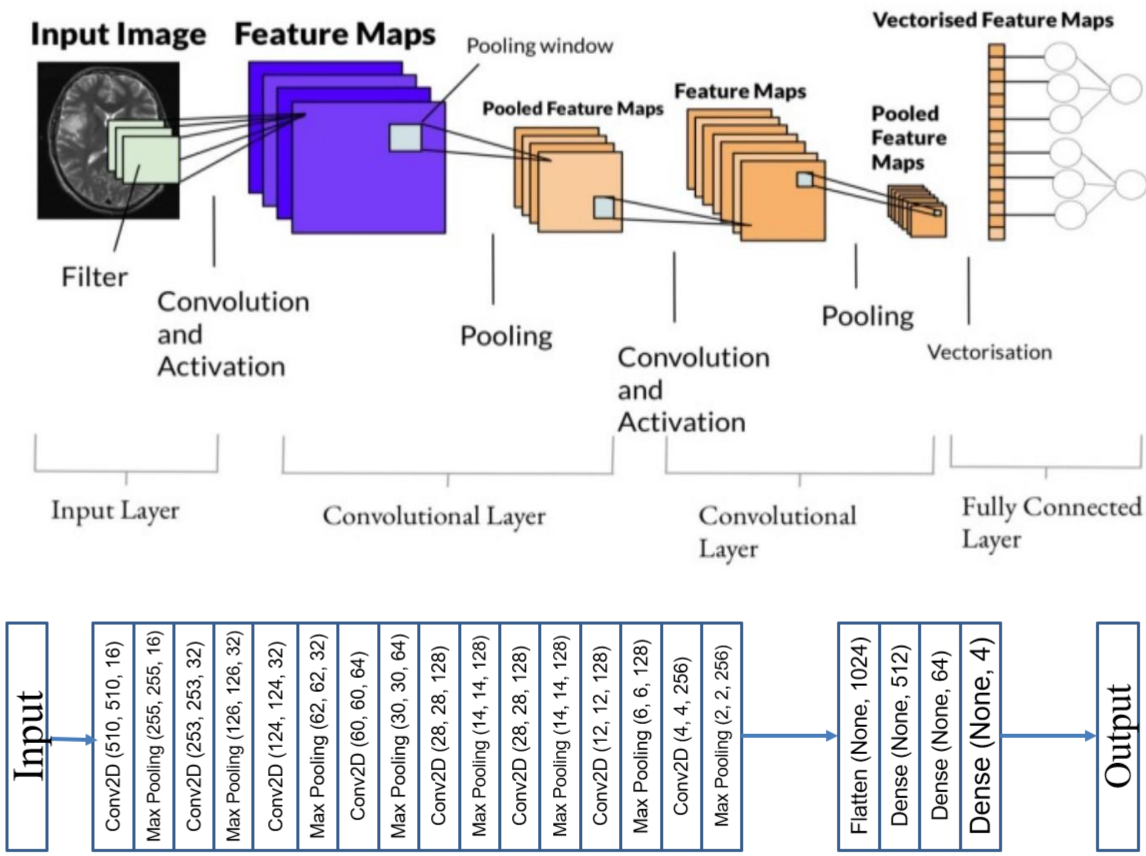


Figure 7.1: CNN architecture

This CNN model is used extensively in modern Machine Learning applications due to its ongoing record-breaking effectiveness. Linear algebra is the basis for how these CNNs work. Matrix vector multiplication is at the heart of how data and weights are represented.

7.1.2 Layers of CNN

Each of the layers contains a different set of characteristics for an image set. For instance, if a face image is the input into a CNN, the network will learn some basic characteristics such as edges, bright spots, dark spots, shapes etc., in its initial layers. The next set of layers will consist of shapes and objects relating to the image which are recognizable such as: eyes, nose and mouth. The subsequent layer consists of aspects that look like actual faces, in other words, shapes and objects which the network can use to define a human face. CNN matches parts rather than the whole image, therefore breaking the image classification process down into smaller parts (features). A 3x3 grid is defined to represent the features extraction by the CNN for evaluation. The following process, known as filtering, involves lining the feature with the image patch. One-by-one, each pixel is multiplied by the corresponding feature pixel, and once completed, all the values are summed and divided by the total number of pixels in the feature space. The final value for the feature is then placed into the feature patch. This process is repeated for the remaining feature patches followed by trying every possible match- repeated application of this filter, which is known as a convolution.

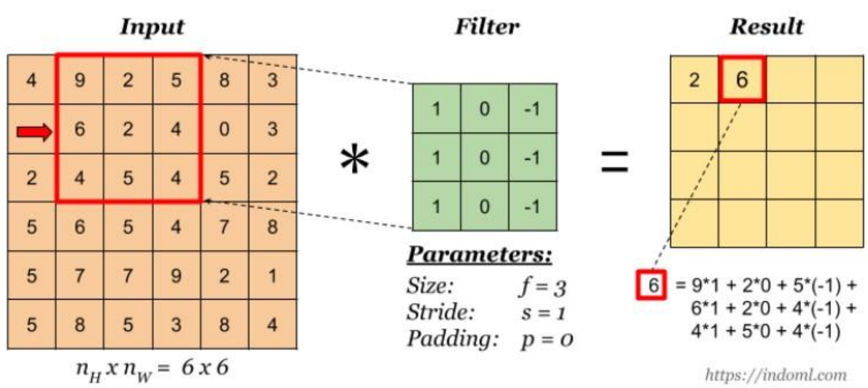


Figure 7.2: Element wise matrix multiplication in convolutional layer.

The next layer of a CNN is referred to as “pooling”, which involves shrinking the image stack. There are two types of pooling: Max pooling and Average pooling. In order to pool an image, the window size must be defined (e.g. usually 2x2/3x3 pixels), the stride must also be defined (e.g. usually 2 pixels). The window is then filtered across the image in strides, with the max value being recorded for each window. Max pooling reduces the dimensionality of each feature map whilst retaining the most important information.

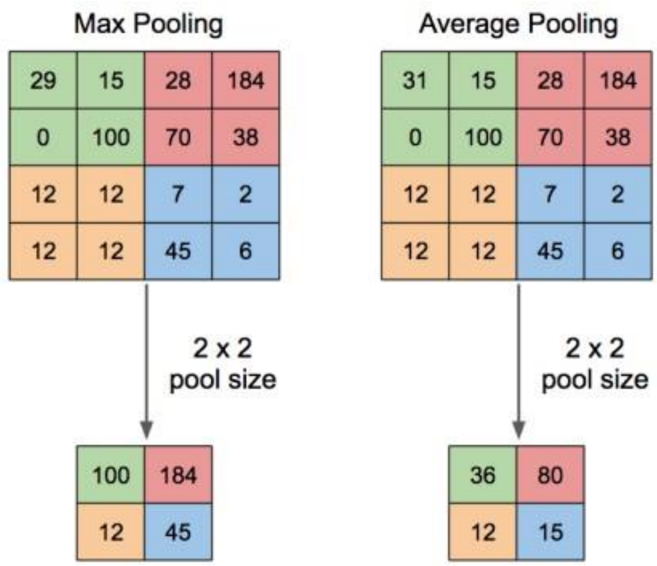


Figure 7.3: Types of pooling

The normalization layer of a CNN, also referred to as the process of Rectified Linear Unit (ReLU), involves changing all negative values within the filtered image to 0. This step is then repeated on all the filtered images, the ReLU layer increases the non-linear properties of the model. The subsequent step by the CNN is to stack the layers (convolution, pooling, ReLU), so that the output of one layer becomes the input of the next. Layers can be repeated resulting in a “deep stacking”.

The final layer within the CNN architecture is called the fully connected layer also known as the classifier. Within this layer every value gets a vote on determining the image classification. Fully connected layers are often stacked together, with each intermediate layer voting on phantom “hidden” categories. In effect, each additional layer allows the network to learn even more sophisticated combinations of features towards better decision making. The values used for the convolution layer as well as the weights for the fully connected layers are obtained through backpropagation, which is done by the deep neural network. Backpropagation is whereby the neural network uses the error in the final answer to determine how much the network adjusts and changes.

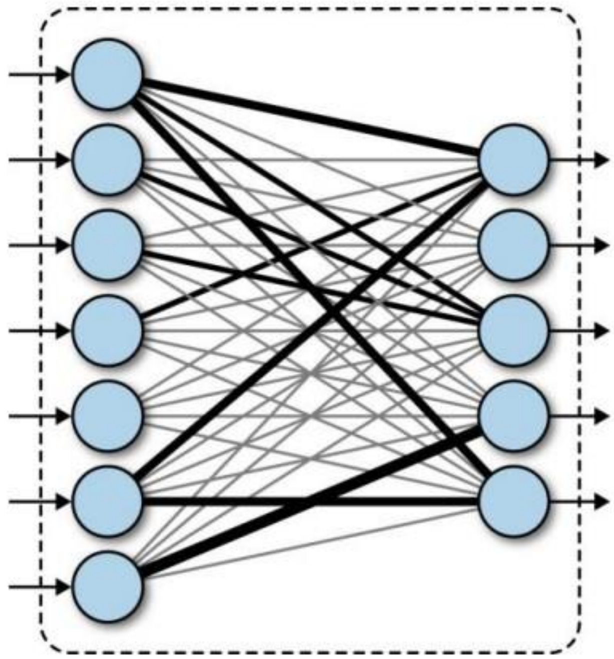


Figure 7.4: Fully connected layer

7.1.3 Model Summary

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 510, 510, 16)	448
max_pooling2d (MaxPooling2D)	(None, 255, 255, 16)	0
conv2d_1 (Conv2D)	(None, 253, 253, 32)	4640
max_pooling2d_1 (MaxPooling2	(None, 126, 126, 32)	0

conv2d_2 (Conv2D)	(None, 124, 124, 32)	9248
max_pooling2d_2 (MaxPooling2	(None, 62, 62, 32)	0
conv2d_3 (Conv2D)	(None, 60, 60, 64)	18496
max_pooling2d_3 (MaxPooling2	(None, 30, 30, 64)	0
conv2d_4 (Conv2D)	(None, 28, 28, 128)	73856
max_pooling2d_4 (MaxPooling2	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147584
max_pooling2d_5 (MaxPooling2	(None, 6, 6, 128)	0
conv2d_6 (Conv2D)	(None, 4, 4, 256)	295168
max_pooling2d_6 (MaxPooling2	(None, 2, 2, 256)	0
flatten (Flatten)	(None, 1024)	0
Retrieval_Layer (Dense)	(None, 512)	524800
dense (Dense)	(None, 64)	32832
dense_1 (Dense)	(None, 4)	260

===

Total params: 1,107,332

Trainable params: 1,107,332

Non-trainable params: 0

7.2 VGG16

7.2.1 What is VGG16?

VGG16 is a CNN model proposed by K. Simonan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". It is a type of CNN (Convolutional Neural Network) that is considered to be one of the best computer vision models to date. The model achieves 92.7% accuracy. The creators of this model evaluated the networks and increased the depth using an architecture with very small (3 × 3) convolution filters, which showed a significant improvement on the prior-art configurations. They pushed the depth to 16–19 weight layers making it approx. 138 trainable parameters.

7.2.2 VGG16 Architecture

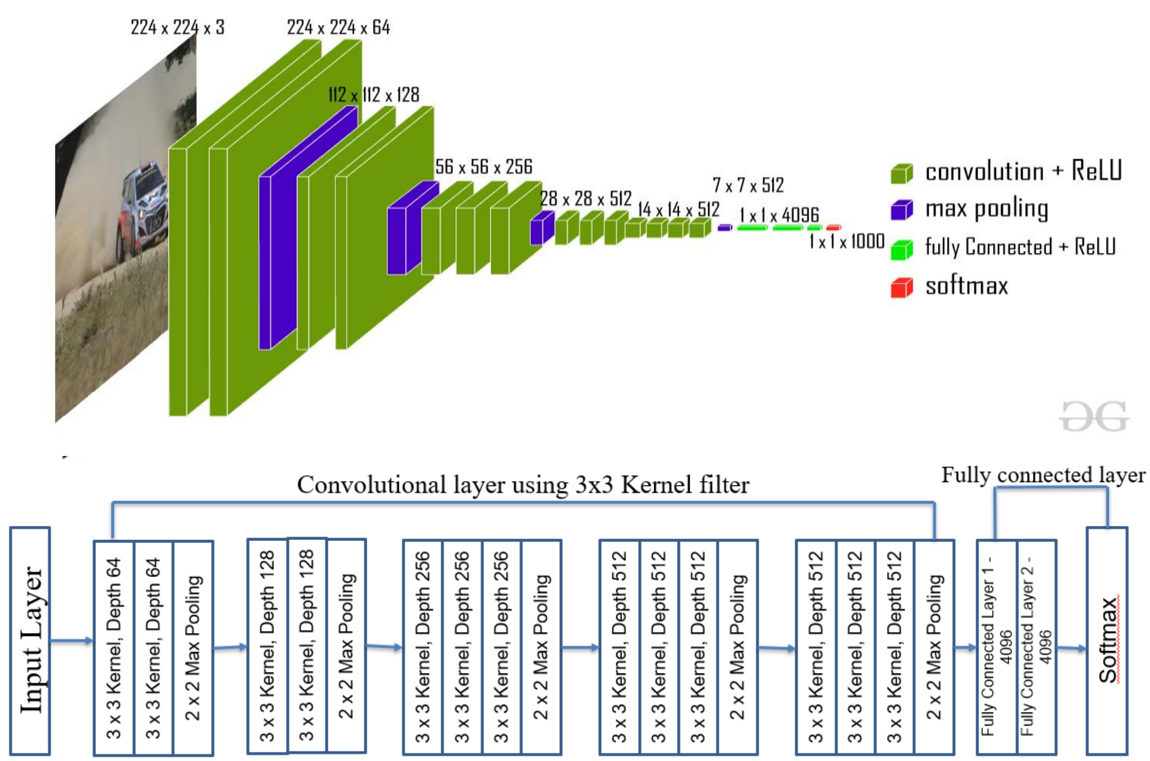


Figure 7.5: VGG16 Architecture

The precise structure of the VGG-16 network shown in Figure 7.5 is as follows:

1. The first and second convolutional layers are comprised of 64 feature kernel filters and size of the filter is 3×3. As input image (RGB image with depth 3) passed into first and second convolutional layer, dimensions changes to 224x224x64. Then the resulting output is passed to max pooling layer with a stride of 2.
2. The third and fourth convolutional layers are of 124 feature kernel filters and size of filter is 3×3. These two layers are followed by a max pooling layer with stride 2 and the resulting output will be reduced to 56x56x128.
3. The fifth, sixth and seventh layers are convolutional layers with kernel size 3×3. All three use 256 feature maps. These layers are followed by a max pooling layer with stride 2.
4. Eighth to thirteen are two sets of convolutional layers with kernel size 3×3. All these sets of convolutional layers have 512 kernel filters. These layers are followed by max pooling layer with stride of 1.
5. Fourteen and fifteen layers are fully connected hidden layers of 4096 units followed by a SoftMax output layer (Sixteenth layer) of 1000 units.

7.2.3 Model Summary

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584

block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
global_average_pooling2d	(Gl (None, 512)	0
dense (Dense)	(None, 1024)	525312

dense_1 (Dense)	(None, 1024)	1049600
dense_2 (Dense)	(None, 512)	524800
dense_3 (Dense)	(None, 2)	1026

===

Total params: 16,815,426

Trainable params: 2,100,738

Non-trainable params: 14,714,688

8. RESULTS AND DISCUSSION

8.1 Accuracy and Loss

The epochs vs accuracy curve plots the training accuracy of the model as a function of the number of epochs. Typically, the accuracy increases as the number of epochs increases, as the model becomes better at fitting the training data. [Figure 8.1.1] shows the Epochs Vs Accuracy between Vgg-16 and Sequential Model.

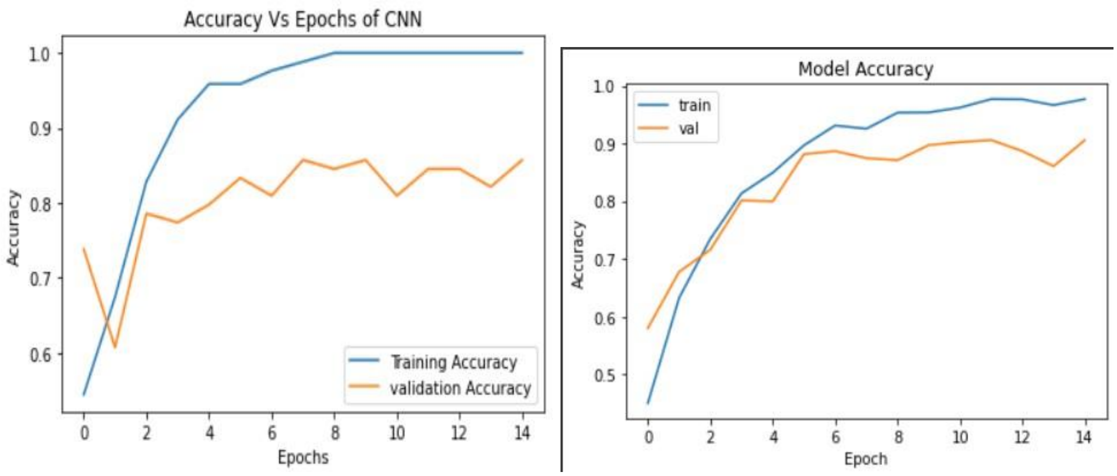
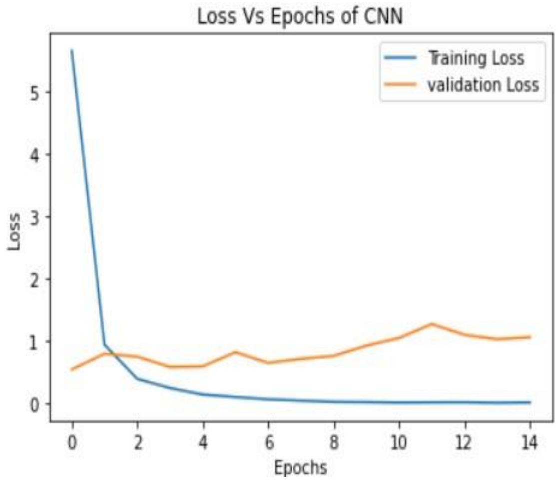


Figure 8.1: Epochs Vs A4ccuracy between Vgg-16 and Sequential Model

The [Figure 8.1.2] shows the Epochs Vs Loss between Vgg-16 and Sequential Model. The loss function measures how well the model is able to make predictions on the training data. As the number of epochs increases, the model is trained on more and more data, which allows it to gradually learn the underlying patterns and relationships in the data. In an ideal scenario, the loss function value should decrease as the number of epochs increases, indicating that the model is improving in its ability to make accurate predictions.



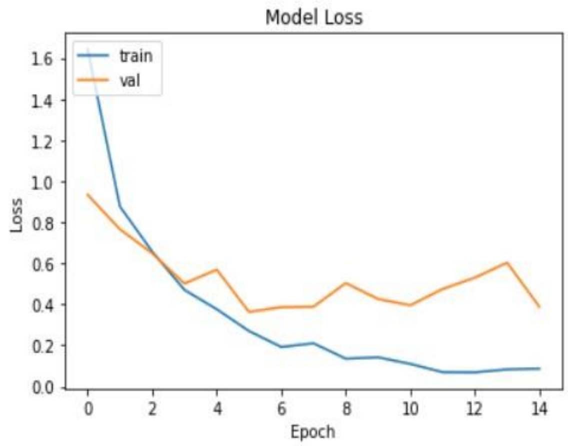


Figure 8.2: Epochs Vs Loss between Vgg-16 and Sequential Model

8.2 Classification Report

The [Figure 8.2.1 & 8.2.2] shows the overall classification report having precision, recall, f1-score for each class.

	precision	recall	f1-score	support
Tumor	0.86	0.76	0.81	33
No Tumor	0.85	0.92	0.89	51
accuracy			0.86	84
macro avg	0.86	0.84	0.85	84
weighted avg	0.86	0.86	0.86	84

Figure 8.3: Classification report for VGG-16

	precision	recall	f1-score	Support
Glioma Tumor	0.92	0.87	0.89	203
Pituitary Tumor	0.97	1	0.99	187
No Tumor	0.94	0.91	0.93	70
Meningioma Tumor	0.85	0.89	0.87	175
accuracy			0.92	632
macro avg	0.92	0.92	0.92	632
weighted avg	0.92	0.92	0.92	632

Figure 8.4: Classification Report Sequential Model

8.3 Confusion Matrix

The [Figure 8.5] is a table that shows the number of correct and incorrect predictions for each class made by a classification model.

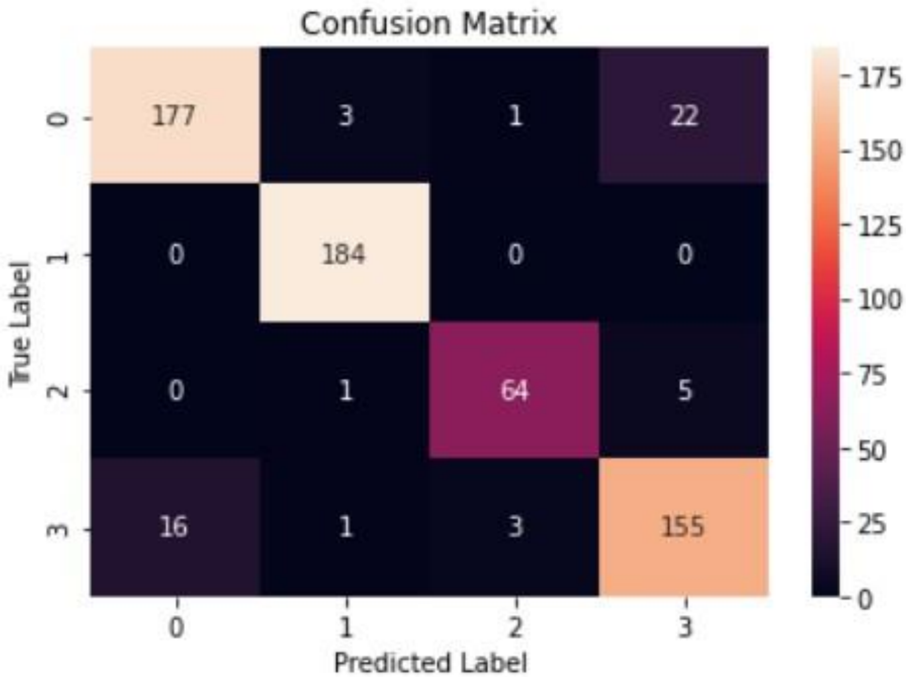


Figure 8.5: Confusion Matrix of Sequential Model

8.4 Receiver Operating Characteristic (ROC) curve:

The [Figure 8.6] is a graphical plot that shows the performance of a classification model at different classification thresholds. It plots the true positive rate (TPR) against the false positive rate (FPR) at different threshold values.

The TPR is the proportion of true positive predictions over the total number of positive cases in the dataset, while the FPR is the proportion of false positive predictions over the total number of negative cases in the dataset.

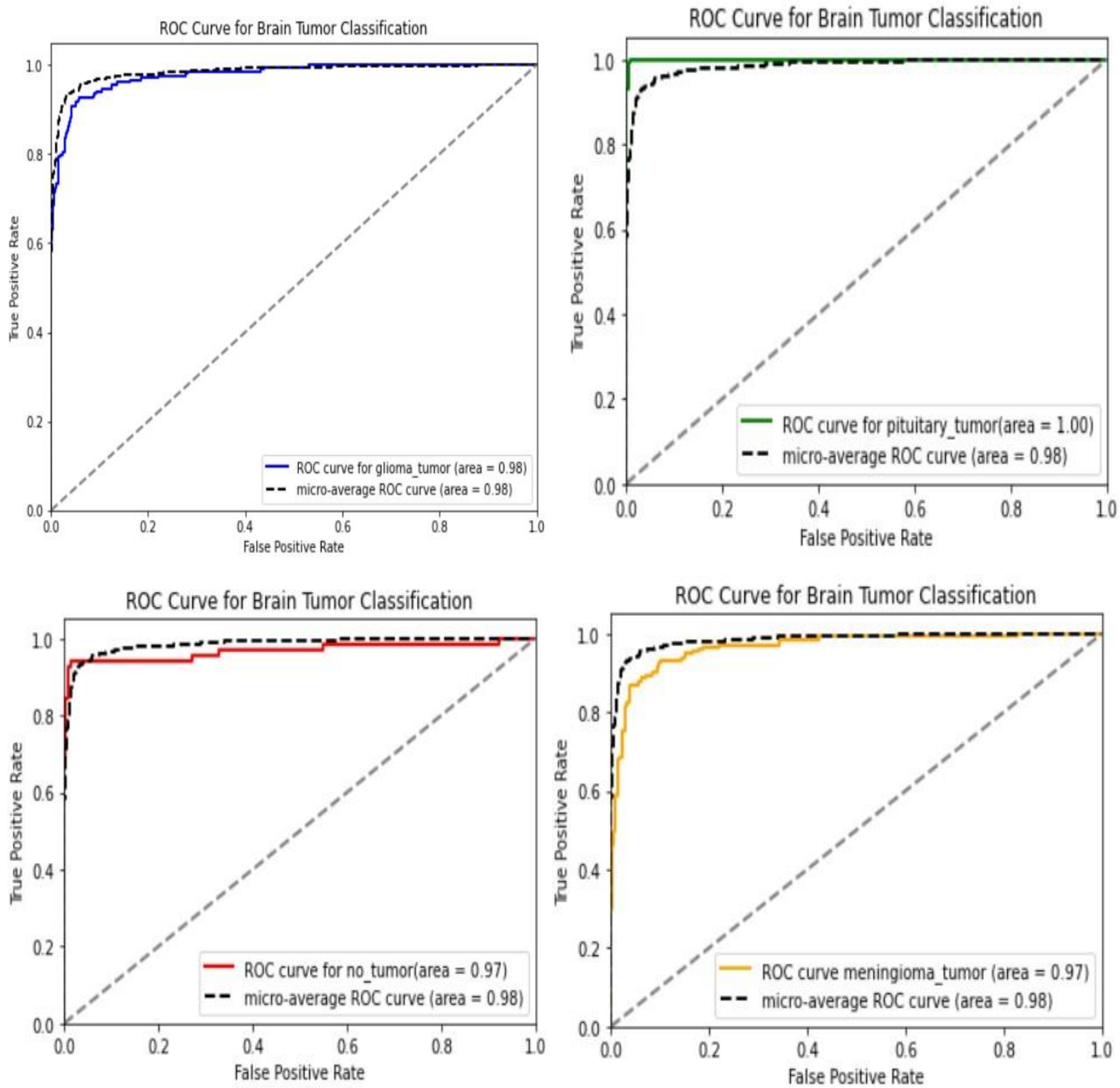


Figure 8.6: Receiver Operating Characteristic (ROC) curve of Sequential Model

8.5 Precision Vs. Recall Curve

The [Figure 8.7] is a plot of precision vs. recall for different classification thresholds. The curve is generated by varying the classification threshold and computing the corresponding precision and recall for each class. The area under the PR curve (AUC-PR) is a summary measure of the overall performance of the model, with a higher value indicating better performance.

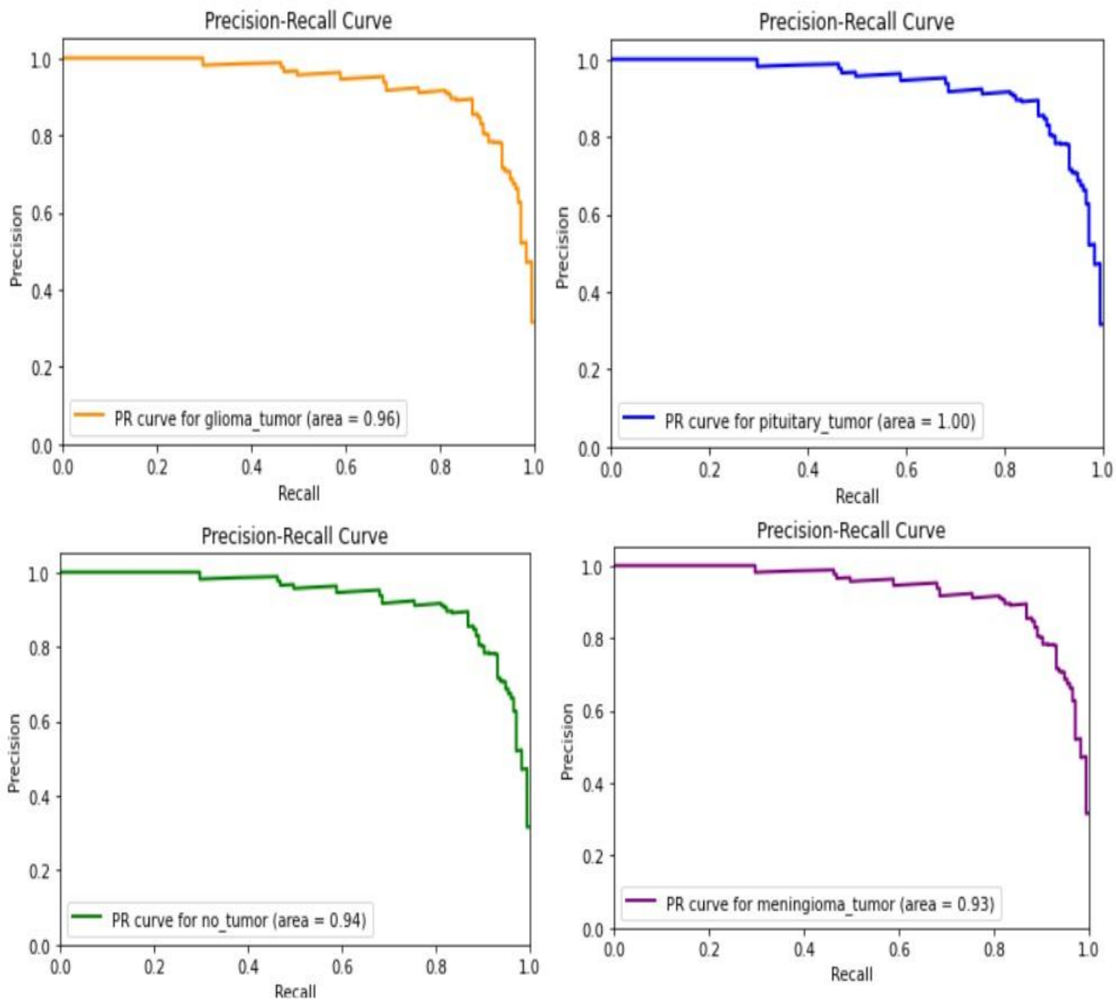


Figure 8.7: Precision Vs. Recall Curve of Sequential Model

8.6 Sensitivity Vs Specificity Vs Accuracy:

The [Figure 8.8] is bar chart which shows Sensitivity, Specificity, Accuracy of the model. Sensitivity, also known as recall, is the proportion of true positives that are correctly identified by the model, specificity is the proportion of true negatives that are correctly identified by the model and accuracy is the proportion of correct predictions over the total.

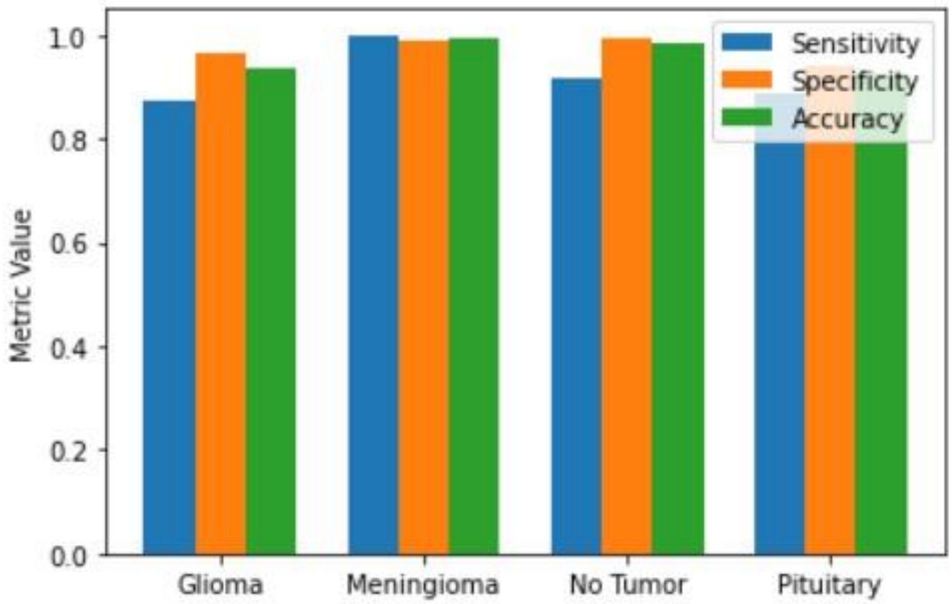


Figure 8.8: Sensitivity Vs Specificity Vs Accuracy of Sequential Model

9. CONCLUSION AND FUTURE WORK

This study presents an accurate and fully automatic system, with minimum pre-processing, for brain tumor classification. The proposed system applied the concept of deep transfer learning to extract features from brain MRI images. The features were used with proven classifier models for an improved performance. Various performance metrics were studied to evaluated model accuracy and ascertain the robustness of the system. The experiment results show the effectiveness of this model despite the smaller amount of training data. The suggested approach can be used for other MRI classification since it requires minimum preprocessing and does not use handcrafted features. As future work, we attend to exploit images from different modalities as T1, T2, and Flair, which aim to augment the dataset size and added robustness to our scheme.

REFERENCES

[1] Deepak, S., and P. M. Ameer. "Brain tumour classification using deep CNN features via transfer learning." Computers in biology and medicine 111 (2019): 103345.

[2] Anushka Singh, Rajeshwari Deshmukh, Riya Jha, Nishi Shahare, Sonam Verma, Prof.Ajinkya Nilawar, Brain Tumor Classification Using CNN and VGG16 Model.

[3] L. Zhou, Z. Zhang, Y.C. Chen, Z.Y. Zhao, X.D. Yin, H.B. Jiang, A deep learning-based radionics model for differentiating benign and malignant renal tumours, Transl, Oncol. 12 (2) (2019) 292–300.

[4] C L Choudhury, Chandrakanta Mahanty, Raghvendra Kumar, B K Mishra Brain Tumor Detection and Classification Using Convolutional Neural Network and Deep Neural Network.

[5] Hemanth, G., M. Janardhan, and L. Sujihelen. "Design and Implementing Brain Tumour Detection Using Machine Learning Approach." In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), pp. 1289-1294. IEEE, 2019.

[6] Ayesha Younis,Li Qiang, Charles Okanda Nyatega, Mohammed Jajere Adamu and Halima Bello Kawuwa, Brain Tumor Analysis Using Deep Learning and VGG-16 Ensembling Learning Approaches.

[7] Smirnov, Evgeny A., Denis M. Timoshenko, and Serge N. Andrianov. "Comparison of regularization methods for imagenet classification with deep convolutional neural networks." Aasri Procedia 6 (2014):89-94.

[8] S. Deepak, P.M. Ameer, Brain tumor classification using deep CNN features via transfer learning.

[9] Wu, Songtao, Shenghua Zhong, and Yan Liu. "Deep residual learning for image steganalysis." Multimedia tools and applications 77, no. 9 (2018): 10437-10453.

[10] Szegedy, C., S. Ioffe, V. Vanhoucke, and A. Alemi. "Inception-ResNet and the Impact of Residual Connections on Learning." arXiv preprint arXiv:1602.07261.